Midterm 2009 – riešenia

1 Formalizmus (10 bodov)

V tejto úlohe klaďte hlavný dôraz na presnosť a korektnosť práce s formalizmom z prednášky.

V tejto úlohe je povolené bez dôkazu použiť len nasledujúce skutočnosti:

- definíciu množiny primitívne rekurzívnych funkcií,
- primitívnu rekurzívnosť nasledujúcich funkcií: $add(x, y) = x+y$, $mul(x, y) = x \cdot y$, $\pi(x) = ((x+1)$. prvočíslo),

$$vydel(x, y) = \begin{cases} x & \leftarrow y = 0 \\ x/y & \leftarrow y > 0 \ \wedge \ y \text{ delí } x \\ x & \leftarrow y > 0 \ \wedge \ y \text{ nedelí } x \end{cases}$$

(Funkcia $vydel$ skúsi hodnotu x vydeliť hodnotou y. Ak sa to bez zvyšku dá spraviť, vydelí ju, inak ju nechá nezmenenú.)

Vyriešte nasledujúce úlohy:

a) (3 body) Dokážte, že $haluz(x, y, z, w) = 2x + y + w$ je primitívne rekurzívna.

b) (7 bodov) Pripomeňme si, že náš kód postupnosti $(a_0, \ldots, a_{n-1})$ je číslo $\pi(n) \cdot \prod_{i=0}^{n-1} \pi(i)^{a_i}$.

Dokážte, že existuje primitívne rekurzívna funkcia $nuluj(k, i)$ s nasledujúcou vlastnosťou: Ak k je platný kód nejakej postupnosti $(a_0, \ldots, a_{n-1})$ pre ktorú $n > i$, tak $nuluj(k, i)$ je kód postupnosti $(a_0, \ldots, a_{i-1}, \mathbf{0}, a_{i+1}, \ldots, a_{n-1})$.

2 Známa pôda (10 bodov)

V tejto úlohe sa pýtam na priamo odprednášané veci, ukážte, že im dostatočne rozumiete.

Zvoľte si model vypočítateľnosti: buď dvojsmerné deterministické konečné automaty s počítadlami, alebo Minského registrové stroje s inštrukciami `INC`, `DEC` a `ZERO`.

Vo zvolenom modeli porovnajte výpočtovú silu strojov so 6 a so 7 počítadlami/registrami.

3 Exkurzia do neznáma (10 bodov)

V tejto úlohe sa pýtam na niečo, čo som neprednášal, a chcem vidieť, ako si s tým poradíte.

V tejto úlohe si zvoľte ľubovoľný vám pohodlný Turing-complete model vypočítateľnosti, ktorý bol spomínaný na prednáške. (Alebo trebárs aj iný, ale potom ho dostatočne popíšte.)

Nezáporné reálne číslo x voláme vyčísliteľné, ak existuje program počítajúci nasledujúcu funkciu f_x:

$$f_x(n) = \begin{cases} \lfloor x \rfloor & \leftarrow n = 0 \\ n\text{-tá desatinná cifra } x & \leftarrow n > 0 \end{cases}$$

Teda napr. $f_\pi(0) = 3$, $f_\pi(1) = 1$, $f_\pi(2) = 4$, atď.

a) (1 bod) Dokážte, že ľubovoľné nezáporné celé číslo je vyčísliteľné.

b) (3 body) Dokážte, že ľubovoľné nezáporné racionálne číslo je vyčísliteľné.

c) (6 bodov) Je $\sqrt{2}$ vyčísliteľná?
Možný hint: Ako je to s vyčísliteľnosťou funkcie $g(x) = \lfloor x\sqrt{2} \rfloor$? A pomôže nám to nejak?

1 Formalizmus (10 bodov) – riešenie

a) Základná myšlienka je, že $2x + y + w = x + x + y + w$, najjednoduchšie teda bude hľadanú funkciu zložiť tak, že spravíme postupne tri sčítania. Pri tom si treba poradiť s tým, aby sme mali správny počet parametrov a sčitovali správne z nich.

V našom riešení najskôr vyrobíme $f(x, y, z, w) = x + x$ a $g(x, y, z, w) = y + w$, a následne ich sčítaním vyrobíme hľadanú funkciu $haluz$:

$$\begin{aligned} f &= Comp[add, P^4_1, P^4_1] \\ g &= Comp[add, P^4_2, P^4_4\ ] \\ haluz &= Comp[add, f, g] \end{aligned}$$

b) Myšlienka: zoberieme kód k a dostatočne veľa krát naň použijeme $vydel$, aby sme vynulovali exponent $\pi(i)$ v k.

Vo funkcii $nuluj$ potrebujeme hodnotu k vydeliť hodnotou $\pi(i)$ presne a_i-krát. Funkcia $vydel$ však má tú príjemnú vlastnosť, že ak y nedelí x, nechá x nezmenené. Preto nepotrebujeme presne zisťovať a_i, stačí nám nejaký jeho horný odhad. Zjavne $k\ \geq \pi(i)^{a_i} > a_i$, preto môžeme ako tento horný odhad použiť hodnotu k.

Pomocou primitívnej rekurzie definujeme pomocnú funkciu $zmensi(n, k, i)$, ktorá hodnotu k skúsi postupne n-krát vydeliť hodnotou $\pi(i)$.

$$\begin{aligned} \pi_4 &= Comp[\pi, P^4_4] \\ g &= Comp[vydel, P^4_1, \pi_4] \\ zmensi &= PR[P^2_1, g] \end{aligned}$$

Teraz do funkcie $zmensi$ vhodne dosadíme, čím dostaneme funkciu $nuluj$. Zjavne $\forall k, i : nuluj(k, i) = zmensi(k, k, i)$, preto:

$$nuluj = Comp[zmensi, P^2_1, P^2_1, P^2_2]$$

2 Známa pôda (10 bodov) – riešenie

V oboch prípadoch pridanie siedmeho počítadla výpočtovú silu nezvýši, oba modely sú Turing-complete. Jedna možnosť dôkazu viedla práve cez ukázanie ekvivalencie oboch s Turingovymi strojmi. Druhá bola priama, ukážeme si tú.

Tvrdenie 1. Pre ľubovoľné $0 \leq a \leq b$ platí, že k ľubovoľnému stroju s a registrami existuje ekvivalentný s b registrami.

Dôkaz: Zjavné, robíme to isté a nadbytočné registre ignorujeme.

Tvrdenie 2: K ľubovoľnému stroju so 7 registrami existuje ekvivalentný s 2 registrami.

Dôkaz: V jednom registri budeme udržiavať hodnotu $2^a 3^b 5^c 7^d 11^e 13^f 17^g$, kde $(a, \dots, g)$ sú aktuálne hodnoty simulovaných 7 registrov. Zvýšiť/znížiť hodnotu simulovaného počítadla vieme násobením/delením príslušnou konštantou, test simulovaného počítadla na nulu tak, že skúsime náš vydeliť príslušnou konštantou, zistíme, či sa to podarilo, a následne vrátime náš register do pôvodného stavu. Pri každej z týchto operácií používame druhý register ako pomocnú premennú.

3 Exkurzia do neznáma (10 bodov) – riešenie

Ako model vypočítateľnosti si zvolíme trebárs náš zjednodušený Pascal. (Len jedna funkcia, ľubovoľne veľké nezáporné celé čísla, atď.)

a) Nech t je nezáporné celé číslo. K nemu zostrojíme nasledujúci program:

```
function f(n : integer) : integer;
begin
   if (n = 0) then f:=t else f:=0;
end.
```

b) Každé nezáporné racionálne číslo t vieme zapísať v tvare $t = a/b$ pre a nezáporné celé a b kladné celé. A ku konkrétnemu a a b vieme zostrojiť takýto program:

```
function f(n : integer) : integer;
var r : integer;
begin
   r := a;
   while (n > 0) do r := (r mod b) * 10;
   f := r div b;
end.
```

Tento program robí presne to, čo my na základnej škole, keď počítame podiel a/b: Pamätáme si aktuálny zvyšok r a dokola: vyrobíme ďalšiu cifru ako celú časť r/b (toto náš program preskakuje, kým nás aktuálna cifry nezaujíma), jej b-násobok odčítame od aktuálneho zvyšku (čím nám ostane hodnota $r \bmod b$) a presunieme sa na ďalšiu cifru, pri čom aktuálny zvyšok prenásobíme 10.

c) Pekne podľa hintu dokážeme najskôr vypočítateľnosť funkcie $g(x) = \lfloor x\sqrt{2} \rfloor$. Túto funkciu počíta napr. nasledujúci program:

```
function g(x : integer) : integer;
var y : integer;
begin
   y := 0;
   while ( (y+1)*(y+1) <= 2*x*x ) do y := y+1;
   f := y;
end.
```

Prečo? Keď hľadáme $\lfloor x\sqrt{2} \rfloor$, hľadáme najväčšie celé y také, že $y \leq x\sqrt{2}$. Uvažujeme len nezáporné x, y, preto po umocnení na druhú dostávame ekvivalentnú nerovnosť $y^2 \leq 2x^2$, a toto už ľahko pre konkrétne y overíme.

V programe jednoducho inicializujeme y na nulu a zvyšujeme ho, kým môžeme. Keďže správna odpoveď je určite nanajvýš rovná x, náš `while`-cyklus po nanajvýš x krokoch skončí.

A teraz si už len stačí uvedomiť, že ľubovoľnú cifru $\sqrt{2}$ vieme vyjadriť pomocou hodnôt funkcie g pre vhodné mocniny 10. Napríklad $g(10^3) = \lfloor 1000\sqrt{2} \rfloor = \lfloor 1000 \cdot 1.414213\dots \rfloor = \lfloor 1414.213\dots \rfloor = 1414$ je číslo tvorené jednotkou a prvými 3 desatinnými ciframi $\sqrt{2}$.

A teda n-tú desatinnú cifru $\sqrt{2}$ vieme vypočítať ako $g(10^n) - 10g(10^{n-1})$.